

Vorlesungsmitschrift

# Semantik von Programmiersprachen

Tobias Höppner

SoSe 2014

## Inhaltsverzeichnis

<b>1</b>	<b>Vorlesung 1 - 17.04.</b>	<b>1</b>
1.1	Was sind Programmiersprachen? . . . . .	1
1.2	Mehrdeutigkeit in natürlichen Sprachen . . . . .	1
1.3	Formalisierungsmethoden . . . . .	1
1.4	Referenzsprache . . . . .	2
1.4.1	Definition der Syntax . . . . .	2

# 1 Vorlesung 1 - 17.04.

## 1.1 Was sind Programmiersprachen?

Programmiersprachen sind künstliche, formale Ausdruckssprachen zur Kommunikation zwischen Mensch und Maschine.

Beim Studium von Sprachen unterscheidet man 3 Ebenen(Aspekte):

- Syntax einschließlich lexikalischer Struktur (Themen des Übersetzerbaus)
  - Kern der Syntax ist die grammatikalische Struktur
  - formale Definition durch kontextfreie Grammatiken
- **Semantik** (diese Vorlesung)
  - Bedeutung
  - Interpretation  
Natürliche Sprachen (Gegenstand der Geisteswissenschaften) lassen Spielräume zur Interpretation offen. Künstliche Sprachen sollen möglichst formalisierbar sein.
  - **Fokus:** Formalisierung
- Pragmatik (Fragen nach dem Gebrauch und Zweck (Useability). *Warum sagt jemand xyz und ist das leicht verständlich?! - Was will jemand damit bewirken?*)

Memo technischer  
Begriff -> z.b. ADD  
reg1 reg2

## 1.2 Mehrdeutigkeit in natürlichen Sprachen

Synonyme : *Schloss, Schimmel, ...*

Auflösung durch Kontext (meist leicht und unproblematisch)

- [Satzebene]: *Dieses Gelände wird zur Verhütung von Straftaten durch die Polizei Videoüberwacht.*

Auflösung durch Hintergrundwissen möglich. weiteres Beispiel *Staatsanwaltschaft ermittelt gegen Betrüger in Clownskostüm.*

## 1.3 Formalisierungsmethoden

Hier werden drei Formalisierungsmethoden für die Semantik von Programmiersprachen behandelt.

### Motivation

- Sicherheit beim Programmentwurf
- Formale verifikation von Eigenschaften
- Richtlinie Übersetzerbau
- Automatische Erzeugung von Programm aus Spezifikation

### Entwicklung der Formalisierungsansätze

- operationelle Semantik (historisch etwas später)  
Man stützt die Bedeutung auf die Funktionsweise technischer und abstrakte Maschinen

ab. Dazu macht man die Maschine so einfach wie möglich und erkläre die Wirkung der Befehle auf die Maschine. (Landin 1964) - Diese Semantik ist ähnlich ähnlich wie die notationelle Semantik (mathematische Notation), jedoch wirklich näher an der Maschine

- **denotationelle Semantik** (etwas früher - fast zeitgleich)  
Formales erfassen durch mathematische Notation. Weitgehende Abstraktion vom Zustandsraum mit einer direkten Zuordnung von syntaktischen Komponenten zu mathematischen Objekten (Semantik). (McCarthy 1962)
- **axiomatische Semantik**  
Veränderung/Transformation von Bedingungen/Prädikaten auf dem Zustandsraum (einer abstrakten Maschine).  
Das geschieht mit mathematischen Formeln. z.B.: Hoareformel: QPR

## 1.4 Referenzsprache

Um alle drei Formalisierungsmethoden zu betrachten nutzen wir die Referenzsprache **WHILE**.

### 1.4.1 Definition der Syntax

(Wie ist die Sprache grammatikalisch aufgebaut?!)

- Elementare Einheiten

---

```
1 // ganze Zahlen
2 Z ::= 0 | 1 | ... | MAX | -1 | -2 | ... | MIN #ZAHL (endlicher Ausschnitt der ganzen
   Zahlen MIN+1 .. MAX)
3 // Wahrheitswerte BOOL
4 W ::= TRUE | FALSE
5 // Konstanten KON
6 K ::= Z | W
7 // Bezeichner bzw. Variablen mit Indizes
8 T ::= a | b | ... | z | a1 | a2 | ... | zi
9 // Operatoren
10 OP ::= + | - | * | / | mod
11 // boolesche Operatoren
12 BOP ::= < | > | = | != | !< | !=
```

---

- Zusätzliche Einheiten (induktiv)  
Die Indizes sind dazu da das Vorkommen von Symbolen in der Struktur eindeutig zu beschreiben.

---

```
1 // Terme TERM
2 T ::= Z | I | T1 OP T2 | read, für T1,T2 in TERM
3 // boolesche Terme BT
4 B ::= W | T1 BOP T2 | read | not B
5 // Befehle (Zustandstransformation) COM
6 C ::= skip | I := T | C1;C | if B then C1 else C2 | while B do C |
   output T | output B
```

---

Warum braucht man für so eine Sprache eine formale Semantik?!

Ich möchte maschinell arbeiten und es gibt Unterspezifikationen, unklar ist das Verhalten bei:

- Typkonflikte
- Fehlerbehandlung

- Rekursion

Syntax ist mehrdeutig?!

---

```
1 while B do C1; C2
```

---

```
1 while B do
2   C1;
3   C2;
```

---

Tabelle 1: main title

Modul	Aufgabe
.	.
.	.

tablenotes here!